



Science Analysis Software & Infrastructure for the Legacy Survey of Space & Time

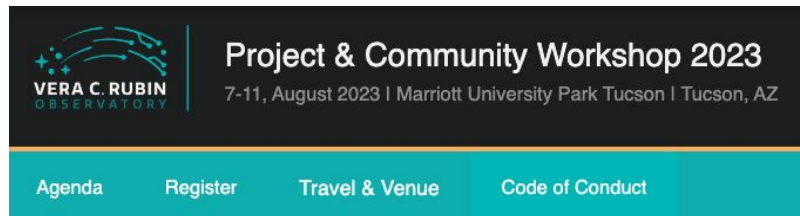
August 9, 2023 11AM Pacific
Canyon ABC & Virtual meeting

Will Clarkson, Andy Connolly, Jeremy Kubica, Rachel Mandelbaum,
Phil Marshall, Knut Olsen, William O'Mullane, Aprajita Verma



U.S. DEPARTMENT OF
ENERGY

Reminder - Code of Conduct



Harassment and unprofessional conduct (including the use of offensive language) of any kind is not permitted at any time and should be reported to:


- Andrew Connolly (ajc@astro.washington.edu),
- John Franklin Crenshaw (jfc20@uw.edu), and/or
- Alysha Shugart (ashugart@lsst.org).




full code of conduct

Rubin Observatory adheres to the principles of kindness, trust, respect, diversity, and inclusiveness in order to provide a learning environment that produces rigor and excellence.


If you feel unsafe at any time send an email to
rubin2023-helpline@lists.lsst.org



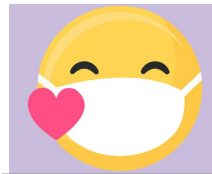
Handshakes OK
Fold Here



Elbow/Fist Bump OK
Fold Here



I Need My Space
Fold Here



Wear a mask if you want to!

Check name-tags for these contact comfort level stickers.

Use the confidential email rubin2023-covid@lists.lsst.org to request a test, report your test results, or ask questions.



If someone is wearing a pin like this, and it indicates a low social battery, please give them their space or offer to restart the conversation at a later time.

Reminder - Virtual Participation



Virtual participants should be muted when they're not speaking.

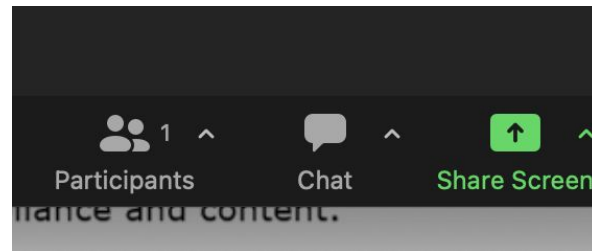


In-person participants should speak into the room microphone(s), or the chair should repeat all questions into the microphone, so that the virtual participants can hear what is said.



In the Rubin2023_PCW Slack Space, all participants can use the session's channel for Q&A and discussion.

The channel name convention is, e.g.:
#day1-mon-slot3a-intro-to-rubin



In Zoom, use the chat to:

- request to unmute to ask a question, or
- type your question so someone can speak it aloud.

The Zoom “raise hand” feature is generally harder for moderators to track, and is not preferred, but may be used at the discretion of the session chair.

Welcome & Intro

- Rubin first light is fast approaching
- S/W infrastructure and analysis are central to our exploitation of LSST-scale data and its complexities
- This session aims to bring all those working on S/W dev to achieve common goals

- Are we ready for data?
 - What infrastructure and interfaces are needed?
 - What has been achieved & what else is needed?
 - Do we have sufficient compute resources?
- Can we work together to achieve common goals?

Agenda

1. Intro (5m)
2. Rubin Data Services - Frossie Economou (10m)
3. TVS S/W Task Force - Alex Razim (10m)
4. DESC: Training scientists to be better software developers - Mike Jarvis (10m)
5. In-kind S/W & Compute resources - Agnès Ferte & Knut Olsen (10m)
6. LINCC Frameworks - Jeremy Kubica (10m)
7. Discussion (30m)

Also have
**Coordination
Groups**
Photo-z
Crowded Field
LSB & forming
cross-matching

Proposal for a Rubin Software Forum

A forum for groups interested in contributing to and engaging in science analysis at LSST scale.

This forum focuses on increasing the visibility of software efforts across the community, and identifying shared infrastructure and educational resources.

By developing effective communication channels for groups developing software for Rubin (both at the Project level and for science analyses), we have an opportunity to create a community with a shared purpose, incentivize collaboration between groups, and develop better software for the research community.

Sharing resources and learning from each other

- Can existing community resources be better advertised or used in a more effective way?
- What lessons have been learned and how can we improve in community software development?

Coordinating efforts

- How do we identify opportunities for collaboration, or identify effort gaps to be filled, across the community?
- Is there appetite for a Rubin S/W forum enabling self-organization within groups that have aligned goals across all LSST SCs? Is that too many organizational structures?



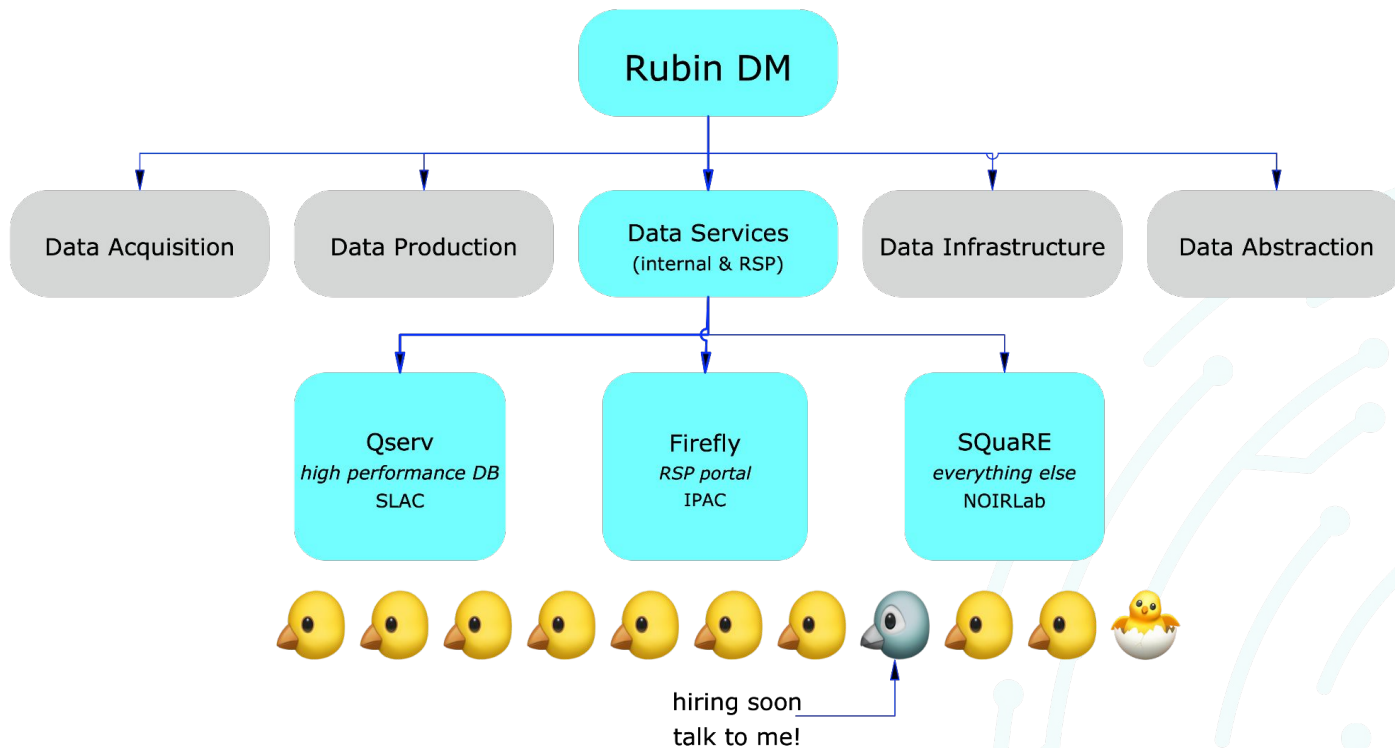
Rubin Data Services

Frossie Economou & Wil O'Mullane



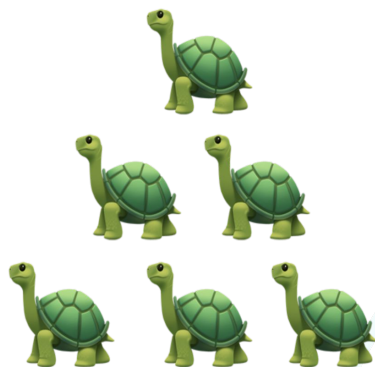
U.S. DEPARTMENT OF
ENERGY

Rubin Data Services



Rubin Science Platform recap

- You Come To The Data, The Data Doesn't Come To You model
 - Despite the length of the project, it's still too much data
- Portal provides guided data discovery and visualisation
- Notebook (Jupyter, terminal, more) provides ad-hoc lightweight analysis and access to services
- API implements VO access protocols (+)
- Unified underlying services inc A&A
- Scaling assumes 10,000 users
 - Based on data rights holders and wide appeal of the data set



RSP Services
science platform

phalanx services
science platform platform

Kubernetes
science platform platform platform

- 15 known deployments
- Work to make deployment more turn-key ongoing

Hybrid model

Hybrid US DAC

Qserv
Butler server
Batch computing
select back-end services
Data stores
Bulk Download Service

SLAC

Portal
Notebook
API
Auth
Isst.io
select back-end services
data cache

Cloud



user

Select thoughts on the challenges of playing in the Rubin software field (and how it can be made to work)

The 10,000 user problem

- The problem is not the computational resources, but the long tail of complex scientific workflows
- Unless you have infinite resources, you need to decide whether you are catering to a _wide_ usecase or a _deep_ usecase.
- RSP is obviously focusing on the widest usecases
 - If you want to go wide, the challenge is to not be duplicative
 - If you want to go deep, the challenge is how to integrate with the wide tools and services, as the deep usecase is likely incremental
- The communication and support challenges are massive at this scale and create painful trade-offs.
- Also, security

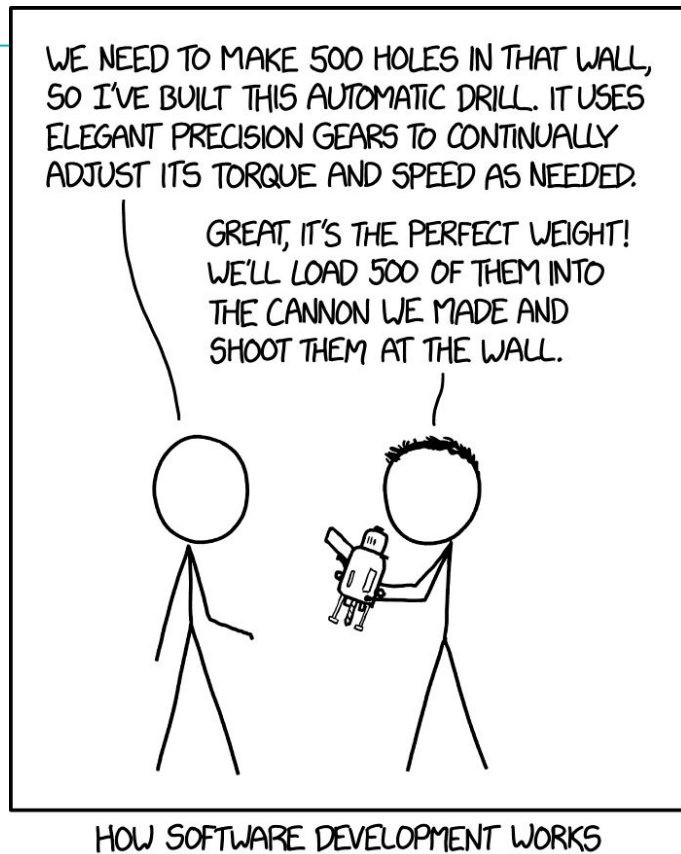


The Grand Old Duke of York

Oh, the grand old Duke of York,
He had ten thousand men;
He marched them up to the top of the hill,
And he marched them down again.
And when they were up, they were up,
And when they were down, they were down,
And when they were only halfway up,
They were neither up nor down.

The collaboration problem

- As a community we have, with a few notable exception, been absolutely terrible at effectively collaborating on software
- ... to the extent that attempts at collaboration can result in negative effort
- Service (in a wider sense) interoperability is a more promising avenue
- But there are significant challenges to achieving this when:
 - Data movement is constrained
 - Software architecture mirrors the funding structure
- Communication bandwidth is a problem
- Also, security



- [illegible]





TVS Software Task Force Overview

Alex Razim on behalf of the TVS Software TF



U.S. DEPARTMENT OF
ENERGY

Software Task Force: organization

- ~40 people from all over the world;
- Bi-weekly meetings, three Software Workshops to date (~annual);
- Main goals:
 - Keep everyone up to date with the software projects in planning or development;
 - Identify, discuss, propose software needed by TVS members;
 - Bring together users and developers from different LSST-related groups (TVS SC, in-kind contributors, RSP developers, LINCC...);
 - Provide (much needed) software development training to astronomers (IEEE Software requirements course, Carpentries Intermediate Research Software Development course);
 - Chisel out recommendations, best practices and workflows for TVS software development;
 - Validate TVS software (make sure it's tested, documented, made available to the final users).

Typical software projects: Dash web-portal

Alex Razim, Lovro Palaversa, Ruder Boscovic Institute, Zagreb, Croatia

Main purposes:

- preliminary investigation of variable sources present in LSST DRs;
- provide a convenient access to the science products, developed within TVS SC;
- serve as a gateway for the server-based software tools, developed within TVS SC;

Planned functionality:

- selection tools (ADQL, pre-defined filters, selection on plots, unsupervised ML clustering);
- sample plotting (scatter, density, histograms);
- light curve analysis (summary about the object, periodogram, folded LC plotting, manual adjustment of parameters such as period and zero point, plotting external data etc.).

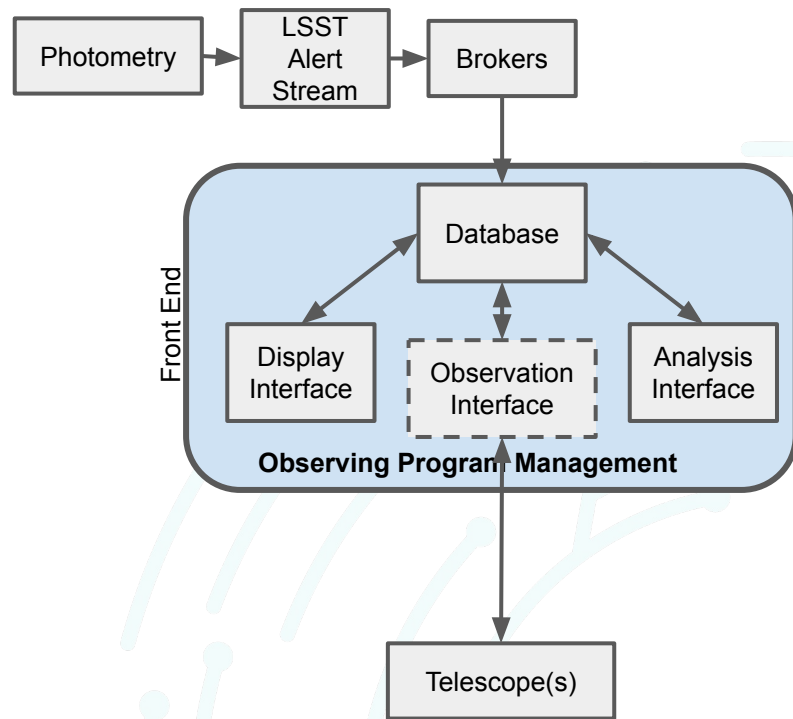
Desired functionality (depends on the resources and collaboration, based on the suggestions from the TVS community):

- period determination (single/batch LCs, multiple algorithms, tuning);
- model fitting;
- classification of variable objects.

Typical software projects: Observing Program Management Tools (OPM)

Yiannis Tsapras & Markus Hundertmark, ARI, Heidelberg, Germany

- What is an OPM system useful for?
 - **Team-specific persistent database** with Rubin alerts and data, augmented with external data & services via APIs and code
 - A way to **manage telescope resources** (e.g. provided as in-kind contribution)
 - Provide **real-time assessment** of ongoing targets of interest and disseminated via APIs
 - **Fully customisable front-end** (you add apps relevant to your science case only)
- An **OPM** is a **Web Service** with all associated advantages
 - Brings together distributed IT resources via APIs
 - Team members can **easily access ongoing projects and data products** (e.g. using mobile devices)
 - It **does not rely on user hardware**. It is a persistent web-service (on the cloud)
 - User customisable - as part of the in-kind contribution we will provide *example* OPMs



Typical software projects: Time domain periodic variability mining pipeline

A. Kovačević, M. Pavlovic, M. Nikolić, L. Popović, D. Ilić, University Belgrade - Faculty of Mathematics, Serbia

Under the hood:

- Conditional neural process modeling of light curves
- Neural network for clustering of light curves
- Time domain extractor of periodic variability
- Statistical robovetters on robustness of periods

Products:

- Numerical & Visual Catalogs of periodic variabilities
- Neural process models of light curves
- Clusters of light curves

PIPELINE APPLICATION RECENT RESULTS

1. [Fatovic, M., Palaversa, L. et al. 2023, Astronomical Journal, 165, 138:](#)

Title: Detecting Long-period Variability in the SDSS Stripe 82 Standards Catalog

2. [Kovacevic, Ilic, Popovic, Mitrovic, Nikolic, Pavlovic, Hajdinjak, Knezevic, Savic, 2023, Universe 9, 287:](#)

Title: Deep Learning of Quasar Light curves in the LSST Era.

Training: Carpentries Intermediate Research software development workshop

- 5 days, 3.5 hour session per day, mostly self-learning in asynchronous mode with online practical sessions
- Organized by: [Software Sustainability Institute](#), Southampton Research Software Group, based on the [Software Carpentry materials](#)
- Curriculum:
 1. **Setting Up Environment For Collaborative Code Development**
Software architecture, Virtual environments, IDE (PyCharm), Version control (PyCharm, Git, Git workflow, Github), Coding style conventions, Linters
 2. **Ensuring Correctness of Software at Scale**
Automatically testing software (Pytest), Scaling up unit tests, Code coverage, Continuous integration (GitHub Actions, Build Matrices), Debugging, Edge testing, Defensive programming, Linters for code robustness
 3. **Software Development as a Process**
Programming vs software development, Software requirements and their types, Software architecture and design, Programming paradigms
 4. **Collaborative Software Development for Reuse**
Code review and pull requests, Software reusability, Code documentation, Licenses, Github tags, Code packaging and dependencies
 5. **Managing and Improving Software Over Its Lifetime**
Collaborative work with GitHub (Issues, Projects, Milestones), Work prioritization (MoSCoW), Sprints

Training: Intermediate Astronomical Software Development course (WIP, Made In TVS)

- Based on the Carpentries Intermediate Research software development workshop;
- Developed by volunteers from the TVS Software Task Force;
- Adjust to astronomers needs:
 - PyCharm -> Jupyter Lab (best practices, widgets/extensions, integration with GitHub, etc);
 - Astronomy-specific code examples;
 - More about software paradigms, comparative examples;
 - More about software documentation;
 - Astronomy-specific testing examples;
 - And so on.

Software validation and best practices

- [Software Requirement](#) and [Software Design](#) documents templates;
- Aiming for proper test coverage and user-testing;
- “TVS-endorsed” Github badge.



DESC

Training scientists to be better software developers

Mike Jarvis



U.S. DEPARTMENT OF
ENERGY

“DESC is a software project disguised as a science collaboration.”

- Phil Marshall

- Almost all of the work we do is writing software.
- Almost none of the students are trained to write software.

∴ We have to train them.

The Monday of every DESC collaboration meeting is called [Dark Energy School](#), where we have 4 pedagogical talks to teach students(*) about various technical topics of relevance to DESC science. [One of these is usually related to software development.](#)

Topics so far:

- Future Computing Architectures and Data Analysis
- Robust Model Fitting with Applications to Astronomy
- Machine Learning in the LSST Era
- Testing Code, as You Code – unit tests and more
- Is my code good enough? Improving software through code review
- Why, when and how to write parallelized code (in DESC)
- Collaborative software development in Python
- Designing libraries and APIs
- Can machine learning solve my problem?
- Robust reproducible results in python — reducing reliance on notebooks!

(*) of all career stages

“The code is more what you’d call ‘guidelines’ than actual rules.”

– Barbossa

To give a more concrete set of suggestions to people who are new to collaborative coding, we assembled a [Coding Guidelines](#) document.

- Guidelines for Coders
 - Getting Started
 - Software Packaging and Licensing
 - Coding Style
 - Documentation
 - Commits
 - Tests
 - Contributing Your Work
- Guidelines for Code Reviewers
 - Who should review
 - Questions to ask
 - What happens after the review
 - Comprehensive Code Reviews
 - Further Reading
- Guidelines for Coding Teams
 - Things to think about
 - Versioning

Software Policy

We tried to make a Software Policy that was minimally authoritarian, but would encourage people to create software that conforms to the following values:

Reliability - produce accurate and reliable results.

Reproducibility - ensure that paper results can be reproduced later.

Reusability - reuse code from previous similar analyses when possible.

Agility - be able to quickly develop code and find bugs quickly.

Flexibility - be able to adapt code to changing analysis choices or science requirements.

Visibility - ensure that developers are given proper recognition and credit.

Trust - trust coding teams to make appropriate choices for their particular software.

Coding teams have broad discretion about the level of testing, documentation, style, etc.

The only requirements are that all software used for a DESC paper

1. **Must be accessible** to all of DESC no later than when the paper is in internal review.
2. **Must be listed in a Software Summar Document**, with authors, versions, licenses, dependencies, what validations were performed, etc.

We encourage all code used for a DESC paper to be open source not later than when the paper is published, but this is not formally required.

Finally, we have a number of avenues for mentorship and more informal guidance of new programmers by more experienced software developers in DESC.

- Regular “hack days” and “sprint weeks” for [focused collaborative coding](#) where people are encouraged to ask for hand-holding and guidance.
- Pipeline scientists are available for targeted software development [guidance and comprehensive code reviews](#).
- Experienced code developers have given numerous [talks about such technical topics](#) as using git, github actions, machine learning techniques, specific DESC tools, etc.

Conclusion

DESC has long been focused on the task of developing high-level software in order to produce the best possible scientific results.

- **Pedagogy**
 - Teach newer programmers the methods of collaborative coding.
- **Guidelines**
 - Provide a concrete list of actionable suggestions.
- **Policy**
 - Set light-handed requirements that incentivize good practices.
- **Mentorship**
 - Help individuals improve their technical prowess at writing good software.



International in-kind software contributions

Agnès Ferté & Knut Olsen



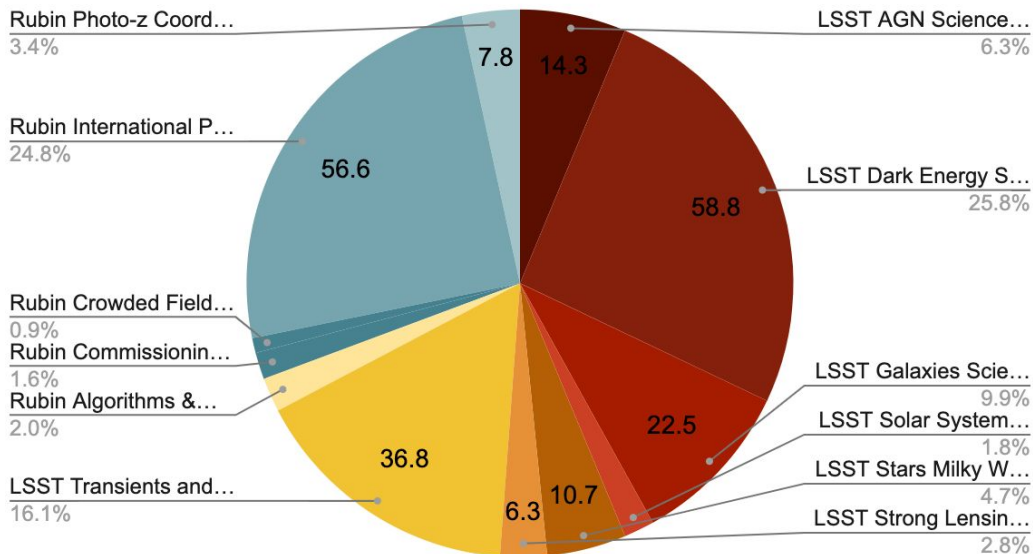
U.S. DEPARTMENT OF
ENERGY

International In-kind Software contributions to Rubin Observatory and Science Collaborations

- Currently 88 from international in-kind Software Contributions:
 - Goal: **develop, document, support software needed to enable LSST.**
 - Recipients are Rubin, Rubin Commissioning, Rubin photo-z & crowded field Coordination Groups, IDACs and LSST Science Collaborations.
- Requirements for softwares developed by in-kind contributors:
 - must be **developed collaboratively**, in a shared, version-controlled, repository that is accessible to the recipient group,
 - **coding standards** are explained in the In-kind Manual <https://ls.st/RDO-041>.

Diversity of S/W Contributions

Software Development Contributions by Primary Recipient



Given the wide scope and diversity of contributions, we would like to enable high return to the community from these S/W contributions and utilise the expertise of the contributors in their respective areas.

Improved connection & coordination to wider S/W efforts and developments across the Rubin ecosystem would help to move towards this.

Software - General Pool: a community resource

The General Pool is made of **professional software developers**:

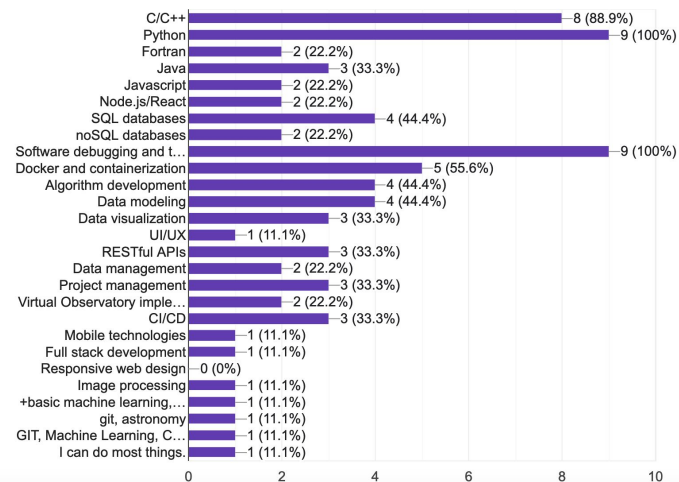
- Goal: resources to **fill in any gaps** in Rubin and SC infrastructure softwares.
- Process: Rubin community submit **proposals** which we review and accept or recommend for resubmission (more details to follow).
- Advantages:
 - **Wide expertise** (continuous integration, testing, web development, etc) to complement astronomers' software skills,
 - **Quick** turn-around,
 - We want the General Pool resources to **support you** so we work with you to further define your proposal if needed,
 - Professional software developers **practices** and potentially **training** on such practices.

Software - General Pool resources in 2023/2024

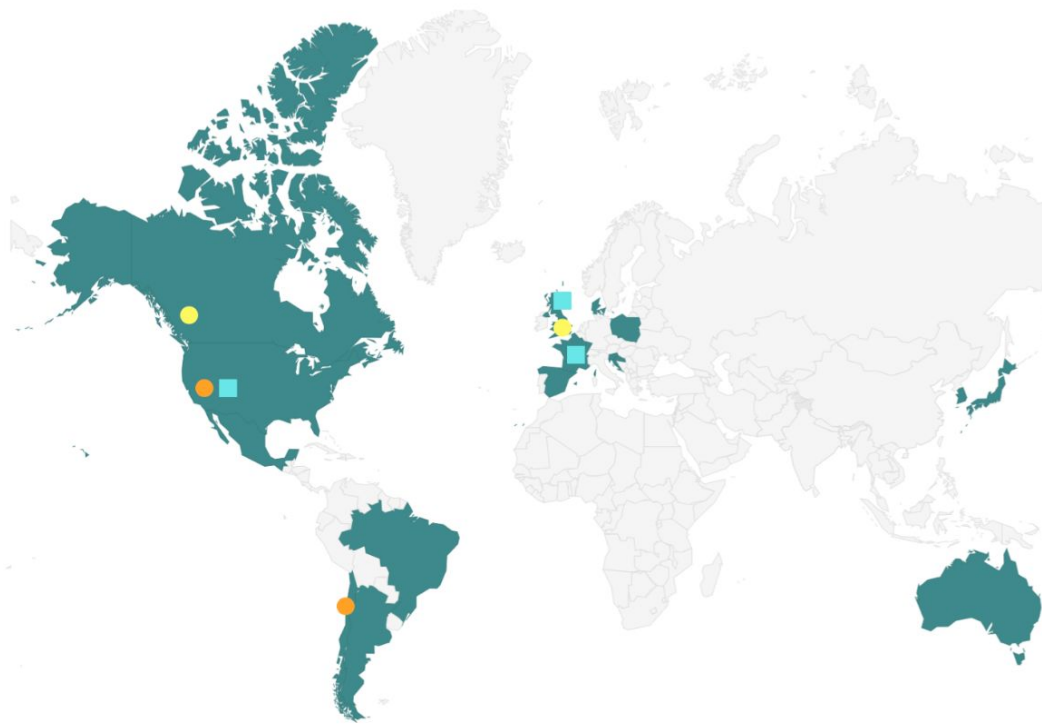
- International expertise:
 - Spain, Hungary, India, Argentina, MPIA...
 - The case of Australia:
 - Astronomy Data And Computing Services (ADACS): software developers dedicated to support astronomy research.

Main contact: Greg Poole

- Define gaps in infrastructure software in your recipient group (Rubin or Science Collaborations).
- Apply at <https://ls.st/ikc-gp>. The **application is always open!**
- We will publish more details on General Pool developers' expertise on the in-kind website soon & hold an information session for prospective applicants.



The Rubin data and computing ecosystem



Data Access Centers	●	Chile, US
Full Independent DACs	●	Canada, UK
Lite Independent DACs		Argentina, Australia, Brazil, Canada, Denmark, Japan (x2), Mexico, Poland, Slovenia, Spain, South Korea
Scientific Processing Center		Croatia
Data Facilities	■	France, UK, US

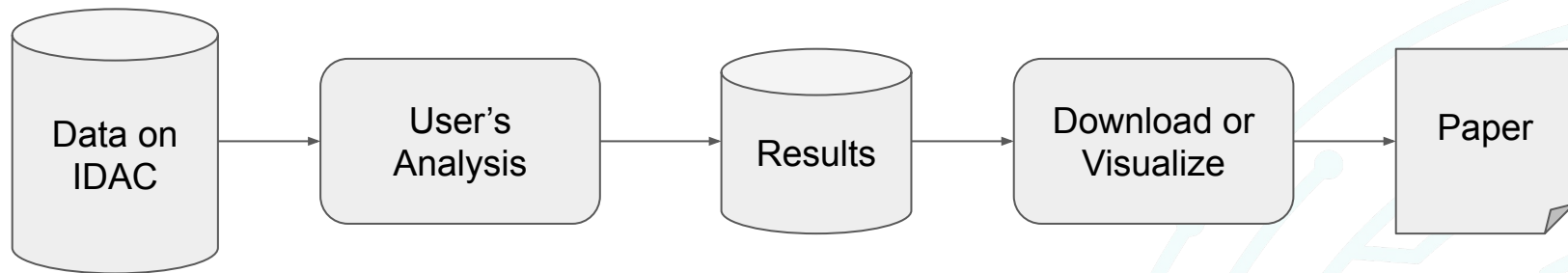
- ~13 Independent Data Access Centers (IDACs)
- Most in-kind computing centers are “Lite”, scoped to store a subset of data
- Geographical distribution is a challenge and opportunity
- Strengths will be the unique capabilities, datasets, and expertise that they can offer
- Use cases are important to understand how resources may be effectively used

IDACs and software

- Resources, compute and storage, over and above the US and Chilean DACs to enhance science for the US community. We are looking to enhance the amount of compute and storage above the Rubin 10% to support science and user generated data products.
- Easily accessed by any data rights holder
- Can be focused on particular science opportunities and collaborations
- Could support other non-LSST data sets
- Support for serving publicly shareable LSST data
- What are the tailored software services each IDAC can provide to maximize science for all with LSST? Workshop held in March 2023 to begin discussion
- Thursday 11:00 - 12:30 session on “Next Steps for In-Kind Computing Resources” for further discussion #day4-thu-in-kind-compute

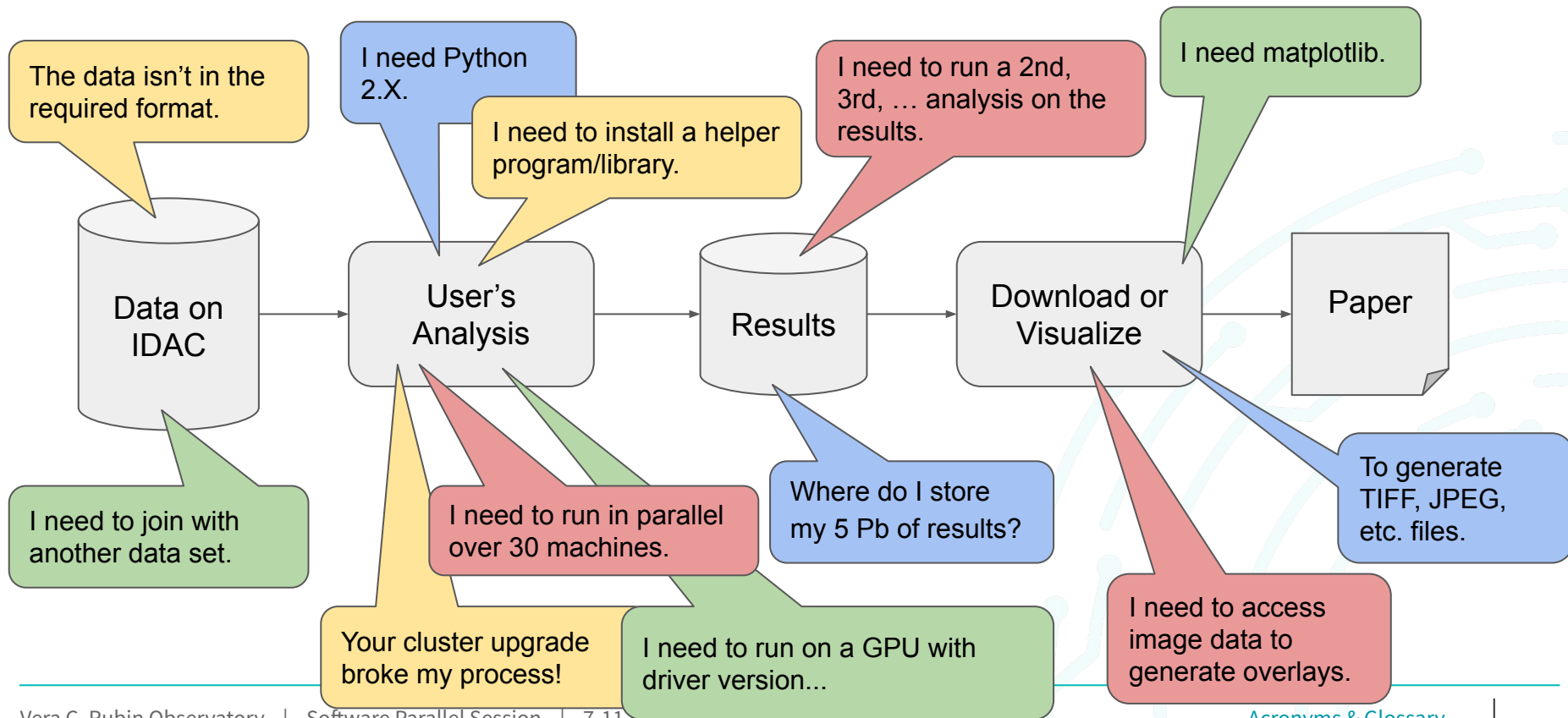
Idealized Software User's Journey

Jeremy Kubica



Real Software User's Journey

Jeremy Kubica





LINCC Frameworks

Jeremy Kubica



U.S. DEPARTMENT OF
ENERGY

The LINCC Frameworks Project

LSST Interdisciplinary Network For Collaboration And Computing

A collaboration between UW, CMU, LSSTC, U Pitt, and NOIRLab to build software, frameworks, and systems for key LSST science.

PIs: Andy Connolly (UW), Rachel Mandelbaum (CMU)

Director of Engineering: Jeremy Kubica (CMU)



<https://www.lsstcorporation.org/lincc/frameworks>

LINCC Frameworks Mission

The LINCC Frameworks team's mission is to enable scientists by developing scalable and productionised software/algorithms in collaboration with broader community.

We want to:

- be engineering and algorithmically focused,
- collaborate with other software efforts (projects may be contributions to existing code bases),
- benefit multiple projects through software development,
- leverage existing tools (build on top of the Rubin Science Platform and standard community tools/libraries), and
- coordinate with community to avoid unnecessary duplication of effort.

- Tech Talks - Talks that showcase the work done by the broad Rubin software and archives community (<https://ls.st/lincc-talks>)
- Workshops - Work with LSST Science Collaborations to identify areas of need.
 - Data to Software to Science Workshop (March 2022)
 - <https://arxiv.org/abs/2208.02781>
- Hackweeks - Provide tutorials and training in any new tools / frameworks.
- Incubators - Scientists work with team to get their science applications working and create software to help the broader community.
- LINCC Frameworks members joining LSST Science Collaborations.

We plan to run three sessions a year (corresponding to the north hemisphere seasons of spring, summer, and fall) with at most two projects per session.

- Each session is ~3 months long
- Applications consist of a 1-2 page proposal (stage 1) and informal follow up conversation (stage 2).

First session is ongoing. Selection for the second session is in progress.

Session	Incubator Dates	Proposal Deadline
Spring	Feb 1st - Apr 30th	Oct 15th
Summer	June 1st - Aug 31st	Feb 15th
Fall	Sep 15th - Dec 15th	June 15th

Dates approximate and subject to change

Supernova Template Fitting for the Age of LSST - PI: Kaylee de Soto (Harvard)

- Scale up SuperPhot+ to handle LSST data stream
- Evaluate computational and accuracy tradeoffs of various models
- Extend the machine learning models to new predictions

Solar System Simulation - PI: Meg Schwamb (Queen's University Belfast)

- Scale up Sorchia (formerly Solar System Post Processing)
- Increase modularity / extensibility
- Conda installable

collaborators: This new project is going to be awesome!

me: I wrote some code, better check it with some tests.

me: Ugh, the test I wrote can't import my module.

me: Fixed that, test runs now - that took longer than expected!

collaborators: Oh! Make it pip installable!

me: huh? Ok, I guess I can figure that out.

me: I keep forgetting to run my tests before I commit.

collaborators: Write some documentation.

me: Ugh, people really use Sphinx? This takes so long to figure out.

me: I made a GitHub action. I hope it runs when it's suppose to.

me: The code seems to work, but don't touch it, it might break.

collaborators: Time for a new project, let's copy this old one as a starting point!

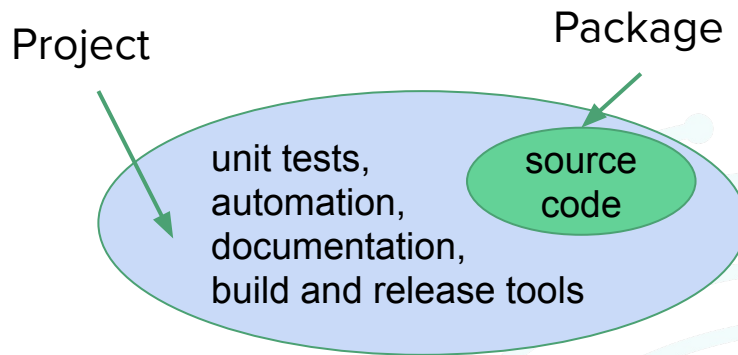
me: There's got to be an easier way.

Why?

- Faster to get a **package** started
- Easier to maintain a **project**
- Easier to collaborate, reduced toil
- Code is easy to build and distribute

Features:

- **Customizable**
- Continuous Integration
- Pre-commit hooks
- Automatic documentation
- Publish to PyPI
- Automatic benchmarking (coming soon)
- Copier as the template tool
- Support for git-lfs, linting, mypy, and more...



Continuous Integration Tests

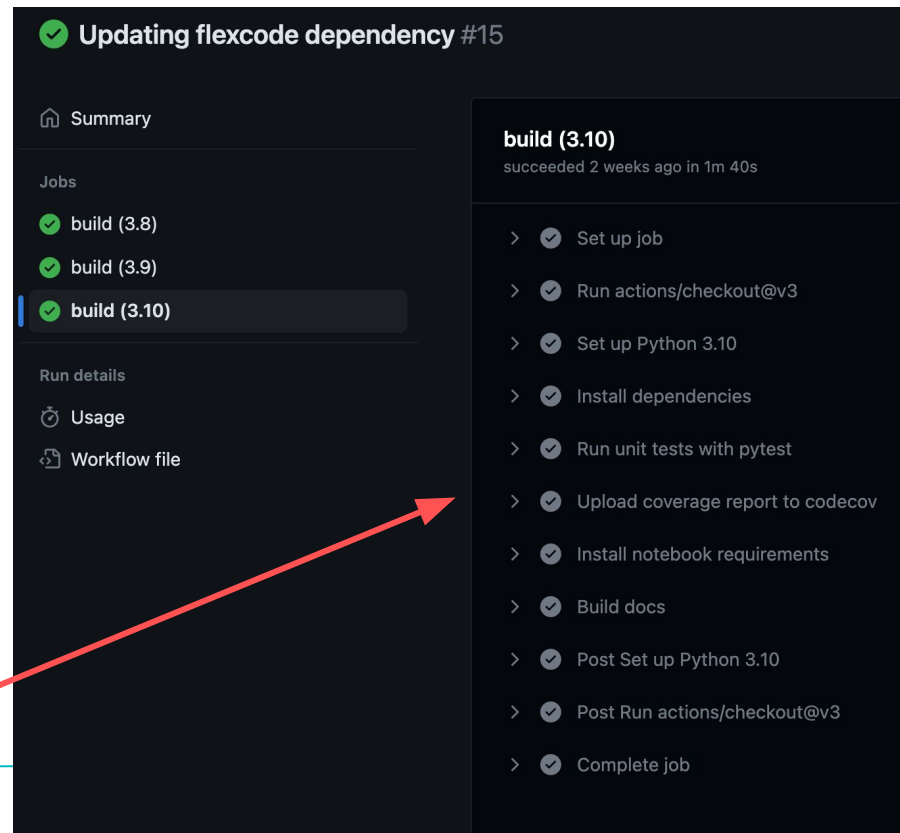
Makes sure that the code runs
in these version of Python.



6 checks passed

- ✓ build
- ✓ build (3.8)
- ✓ build (3.9)
- ✓ build (3.10)
- ✓ codecov/patch Coverage not affected when comparing f4470c0...7dc2786
- ✓ codecov/project 98.63% (+0.00%) compared to f4470c0

Each build executes
each of these steps.



✓ Updating flexcode dependency #15

Summary

Jobs

- ✓ build (3.8)
- ✓ build (3.9)
- ✓ build (3.10)

Run details

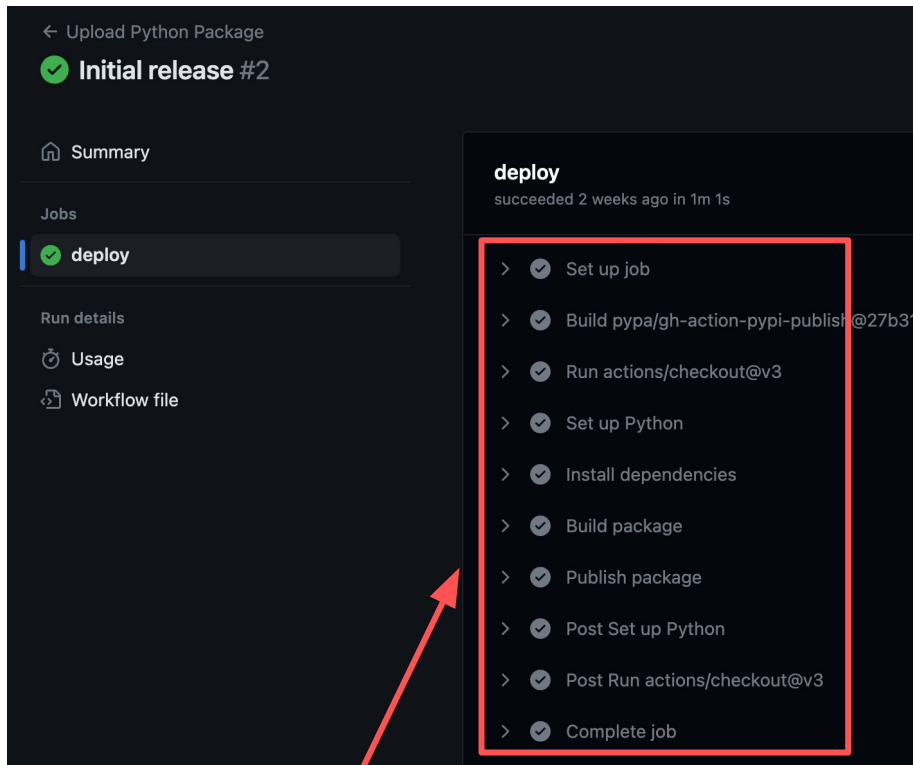
- Usage
- Workflow file

build (3.10)
succeeded 2 weeks ago in 1m 40s

- > ✓ Set up job
- > ✓ Run actions/checkout@v3
- > ✓ Set up Python 3.10
- > ✓ Install dependencies
- > ✓ Run unit tests with pytest
- > ✓ Upload coverage report to codecov
- > ✓ Install notebook requirements
- > ✓ Build docs
- > ✓ Post Set up Python 3.10
- > ✓ Post Run actions/checkout@v3
- > ✓ Complete job

Automatically Publish to PyPI

Credit: Drew Oldag



← Upload Python Package

✓ Initial release #2

Summary

Jobs

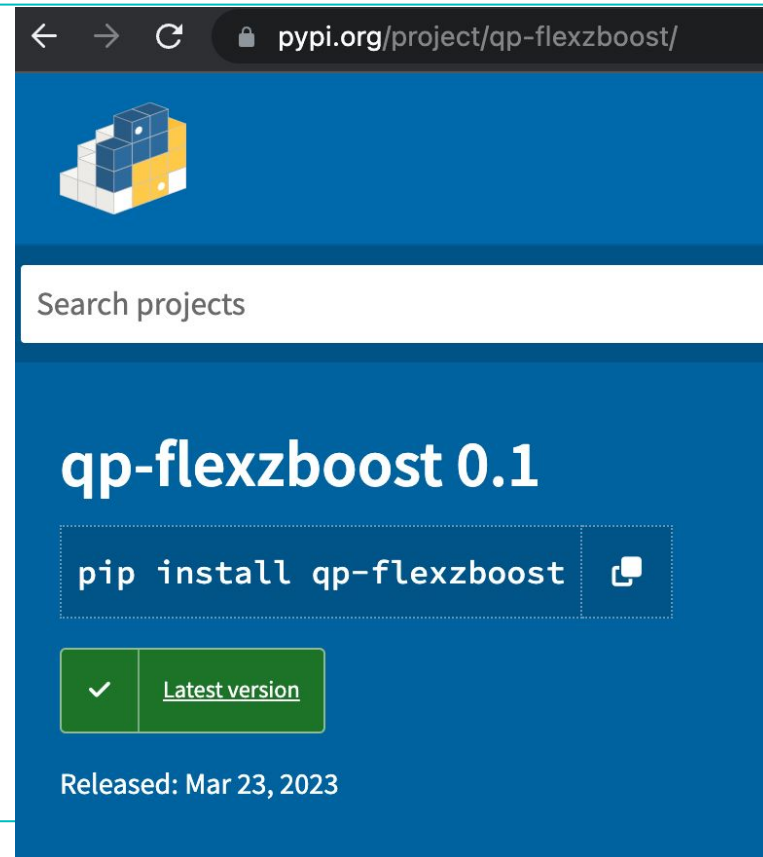
- ✓ deploy

Run details


- Usage
- Workflow file

deploy
succeeded 2 weeks ago in 1m 1s

- ✓ Set up job
- ✓ Build pypa/gh-action-pypi-publish@27b31
- ✓ Run actions/checkout@v3
- ✓ Set up Python
- ✓ Install dependencies
- ✓ Build package
- ✓ Publish package
- ✓ Post Set up Python
- ✓ Post Run actions/checkout@v3
- ✓ Complete job




← → ↺ pypi.org/project/qp-flexzboost/



Search projects

qp-flexzboost 0.1

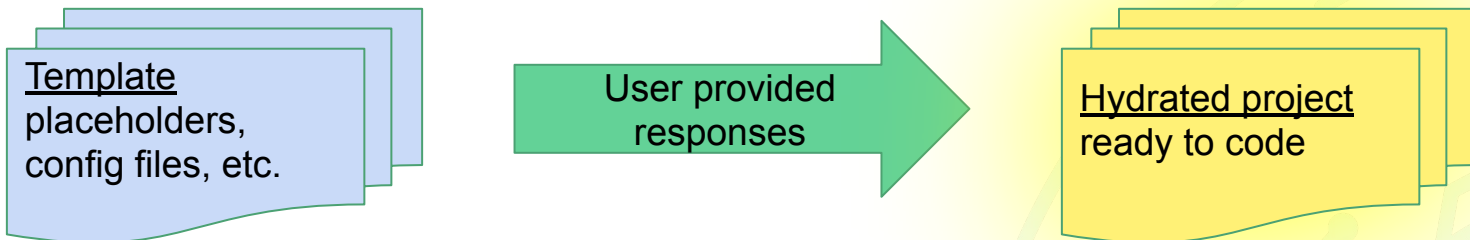
`pip install qp-flexzboost` 

✓ Latest version

Released: Mar 23, 2023

<https://github.com/lincc-frameworks/python-project-template>

- Copy/paste an old project to a new folder 😡
- Clone a simple package from a repository 😞
- Cookiecutter 😐
- Copier 😊



For more information see the tech talk:

<https://www.youtube.com/watch?v=nHHG-sZRRUg>

LINCC Frameworks - Summary

- Goal is to enable science on Rubin's LSST through software development.
 - Collaborators welcome!
 - Feedback on areas of focus welcome.
- Near term opportunities:
 - Alpha test software (LSDB, TAPE, etc.)
 - Adopt or contribute to the python project template.
 - Watch or give tech talks
 - Apply for incubators.
 - Reach out about other collaboration opportunities.



Discussion



U.S. DEPARTMENT OF
ENERGY

Sharing resources and learning from each other

- Can existing community resources be better advertised or used in a more effective way?
- What lessons have been learned and how can we improve in community software development?

Coordinating efforts

- How do we identify opportunities for collaboration, or identify effort gaps to be filled, across the community?
- Is there appetite for a Rubin s/w forum enabling self-organization within groups that have aligned goals across all LSST SCs? Is that too many organizational structures?



Thanks!



U.S. DEPARTMENT OF
ENERGY